OOD Assignment 2 – 80 Points – UML/Java OO System Design and Development

This introductory assignment is intended to be done by you as a solo effort. The goal of the is to prepare you to perform OO design and development tasks with UML and Java. The design and code you develop here will be used as the starting point for implementing patterns and enhancements in assignments 3 and 4 as well. You should use a Java 8 or later environment to develop this code (I recommend Java 17).

Overall Assignment – Zoo Simulation

We will simulate a zoo in this assignment, focusing on animals and zoo staff members. The Zoo contains a variety of Animals. Three families of animals are provided for you. You should add a fourth of your choosing. In your solution to modeling the Zoo, you should represent the Animals in a class inheritance hierarchy as follows (Animal – Family – Type):

Animal

Pachyderm

Rhino, Elephant, Hippo

Feline

Tiger, Lion, Cheetah

Birds

Parrot, Falcon, Owl

[Custom Animal Family you select]

[Three Animal Types you select]

An Animal has attributes including Name (a string), Size (an enumerated value – small, medium, large, extralarge), and Healthy (a boolean). Animals have methods including sleep, roam, eat, makeSound.

- Each method when called should issue a print statement saying the action has been taken by the specific instance of the animal, for instance:
  - Tammy the Tiger has gone to sleep.
  - Rita the Rhino is roaming – and charged!
  - Cory the Cheetah is eating 0 food units.
- sleep is a common method to all the animals.
- roam varies at the family level.
  - Pachyderms may charge during a roam (25% of the time).
  - Felines may sleep instead of roaming (50% of the time).
- eat is common to all animals. There is a 10% chance an animal eats no food when fed, and a 10% chance they eat too much.
- makeSound is unique to each type of animal.

The Zoo is made up of Enclosures and a Hospital. There is an Enclosure for each Type of Animal. Enclosures are named for the Type of Animal kept in them. Animals are assigned to an Enclosure if Healthy or to the Hospital if

not Healthy. You will need to create appropriate associations to relate Enclosures and the Hospital to the Zoo, and Animals to Enclosures (by Type) or the Hospital.

The Zoo also has Staff. Currently there are two subclasses of Staff, Handlers and Veterinarians. The Zoo will have a Handler assigned to each Family of Animals, and a single Veterinarian assigned to the Hospital.

The Zoo simulation will run for 30 days. There are three unique instances of each Animal Type in all the Enclosures at the beginning of the simulation (and no Animals in the Hospital). Display the start and end of each numbered day in the console.

In a given day, the Staff will perform the following actions.

1) wakeAnimals - Each Handler will wake their assigned Animals – when awakened, Animals will makeSound.
2) feedAnimals - Each Handler will feed their assigned Animals – each Animal will perform an eat action. If an Animal does not eat or eats too much, there is a 50% chance of becoming not Healthy. If the Animal is not Healthy it is moved from the Enclosure to the Hospital (this should be announced).
3) zooStatus - At this point the Zoo opens for the day, you should display a tabular list of the individual Animals in each Enclosure along with the name of the Handler and a list of individual Animals in the Hospital along with the name of the Veterinarian.
4) exerciseAnimals - Each Handler will exercise their assigned Animals – each Animal will perform a roam action.
5) treatAnimals - The Veterinarian will treat sick Animals. Each Animal in the hospital will have a 50% chance of becoming Healthy. If they become Healthy, they will be moved to their normal enclosure.
6) bedAnimals - Each Handler will bed down their Animals for the evening – each Animal will perform a sleep action.

When Staff or Animals perform actions, the appropriate print statements should be sent to the console. For Staff, this may look like:

> Felix the Feline Handler is exercising animals.
>
> Valerie the Vet is treating sick animals.
>
> Tony the Tiger is feeling better and is moving back to its enclosure.

An approach will need to be created to find unique names for Animals and Staff (the names can be unique strings, strings with numbers, etc.). This naming method should be used when the Animal or Staff object is constructed.

Assignment 2 Part 1 – UML designs for the Zoo Simulation

Using UML tools discussed in class, create the following UML diagrams.

1. A UML Class Diagram – This UML class diagram should include all classes, abstract classes, or interface definitions you feel you should use to design the Zoo simulation. The diagram should be complete, modeling all classes in use in the application. The diagram should clearly indicate any aggregation, association, or inheritance relationships, including any required multiplicity. In class elements of the diagram, identify key attributes and methods, including accessibility. (Usually this does not include constructors.)

2. A UML Activity Diagram – Create a UML activity diagram that shows the flow of the application. This includes initialization and the activities of Animals and Staff in a typical day. This can be generalized to refer to the actions of Staff and Animals, and not unique individuals. Do show the path that Animals may take to go in or out of the Hospital.

Please include your name on the submitted materials.

Submission Guidelines – Part 2

I would prefer that the diagrams are made with an electronic drawing tool for clarity. If you must hand draw the diagram (on a whiteboard or on paper) it must be complete and legible. The diagram should be turned in as a PDF. Most tools you use can print their output to a PDF, if necessary, you can break large diagrams into two or more pages. The important part is that what you turn in is readable, and remember to put in your name!

Assignment 2 Part 2 – Java Code for the Zoo Simulation

Carefully consider the construction of your Java code based on your OO Design work. Implement the code required in Java to create, initialize, and run the Zoo simulation for 30 days. The code you write for the simulation should show examples of the following OO mechanisms: **inheritance, polymorphism, delegation, cohesion, and aggregation/composition**. Clearly comment on at least one example of each of these in your code so that they stand out when code is reviewed.

In addition, all code should be appropriately commented. Any .java files should include a *header block* with the author's name, the class, the program purpose, and the last revision date in the comments. Each *class/interface* and significant *method* in the code should be commented on as well. You do not have to use Javadoc formats for these comments, but you certainly can.

Capture ALL output from the simulation to the console (by cut/paste to a text file or optionally by directly writing to a text file). Also, update the UML class diagram from Part 1 to represent the final outcome of your Java code. Note any changes on the diagram since Part 1 as an annotation on the diagram itself.

If you use code in whole or in part from any external source, say Stack Overflow for example, you must cite your original source and include the URL in your comments. Never submit code copied from another source without providing appropriate citations.

Code in the Java main function should be limited to basic object instantiation and execution. No large blocks of procedural code or logic should be in Java main. All operations and information should be handled between appropriate objects of the solution classes.

Submission Guidelines – Part 2

Submit the following files to CougarVIEW for Part 2:

- any .java files required to run your program
- one text file with captured output from running the program for 30 simulated days
- your updated UML class diagram from Part 1 (as a PDF)

Grading Rubric:

Assignment 2 is worth 80 points total.

Part 1 (UML Diagrams) is 30 points and is due on Friday September 8 at 11:59 PM

- Each diagram is worth 15 points. A thorough design that meets all the requirements for class content and UML annotations will earn full points. Severe misses in UML syntax or incomplete design implementations will be penalized.

Part 2 (Java Code & Results) is worth 50 points and is due on Friday September 15 at 11:59 PM

- (15 points) The code will be reviewed for good OO design – cohesive classes with clear logical connectivity and well named methods. Procedural elements or code duplication will be penalized. Code should be thoroughly commented as requested.
- (10 points) The five examples of OO concepts (**inheritance, polymorphism, delegation, cohesion, and aggregation/composition**) should be identified in comments.
- (15 points) The output from code execution should show all expected object actions and flow. Missing elements will be penalized.
- (10 points) The updated UML Class Diagram should clearly show any changes since part 1, and an annotation on the drawing should describe the major changes. If no changes were made, this should be clearly annotated on the drawing as well.

Overall Project Guidelines

Assignments will be accepted late per the published policy from the syllabus:

- Up to two days late from posted submission date/time: -5% grade penalty
- Up to an additional three days late from posted submission date/time: -15% grade penalty
- Assignments will not be accepted after 5 days from posted submission date/time

Use office/student hours and e-mail to reach the instructor regarding homework/project questions, or if you have issues in completing the assignment for any reason.